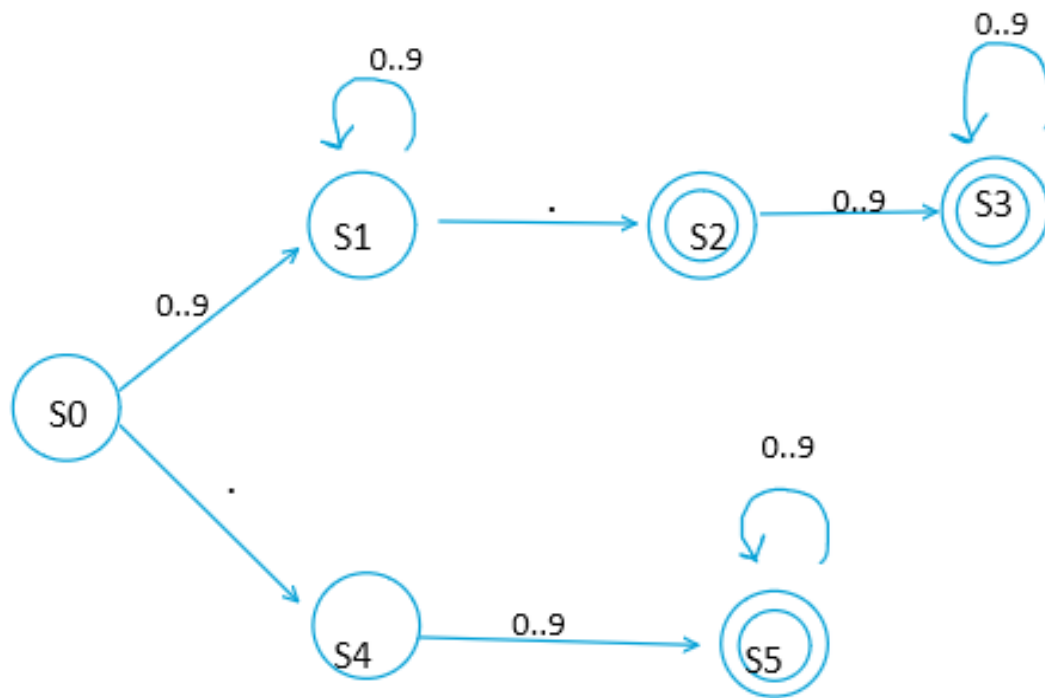


Μεταγλωττιστές 2020

Προγραμματιστική Εργασία #1

Ονοματεπώνυμο: Πολίτης Αλέξανδρος

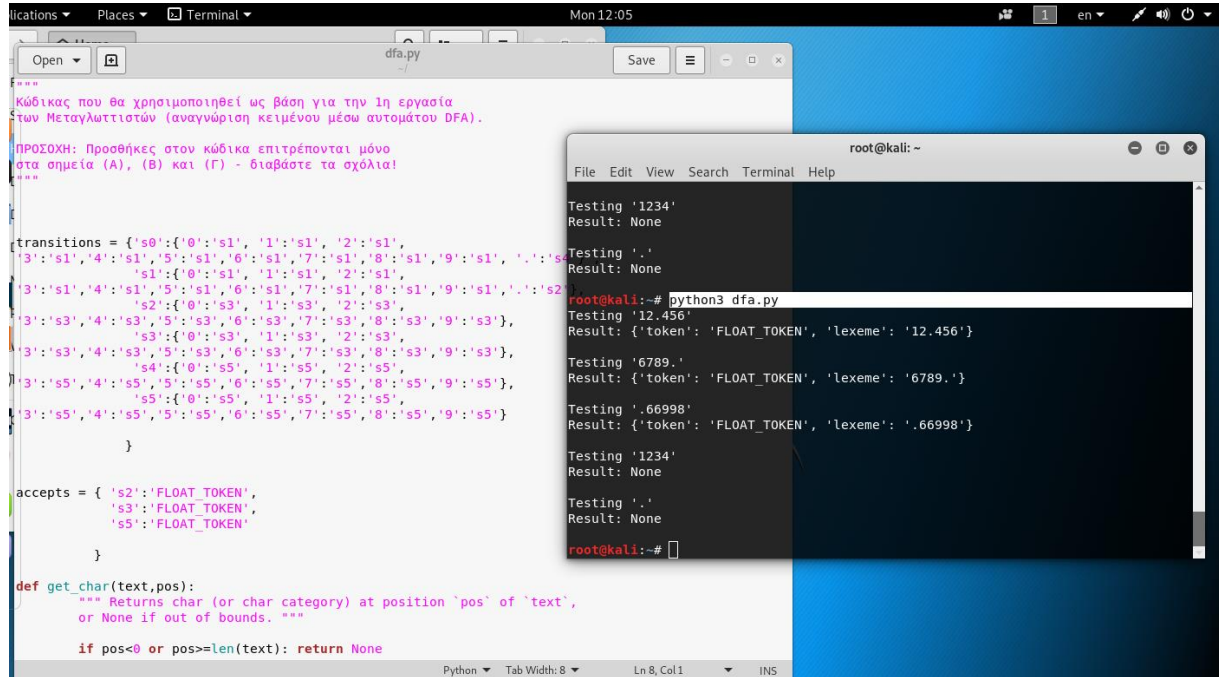
A.M.: Π2017202



Αυτόματο για την αναγνώριση κλασματικών αριθμών

Το συγκεκριμένο αυτόματο πεπερασμένων καταστάσεων (FA) αναγνωρίζει κλασματικούς αριθμούς στις μορφές: 123.56 , 123. , .56789. Ωστόσο, δεν αναγνωρίζει αριθμούς χωρίς την υποδιαστολή πχ. 1234 αλλά ούτε και την τελεία μόνο.

# Αποτελέσματα εξόδου του προγράμματος



The image shows a code editor window with a file named `dfa.py` and a terminal window running the script. The code defines a DFA with states `s0` through `s5` and transitions based on digits 0-9. It also defines a set of accepting states `accepts` containing `'s2'`, `'s3'`, and `'s5'`. A function `get_char` is defined to return the character at a specific position in a string, or `None` if out of bounds.

```
'''
Κώδικας που θα χρησιμοποιηθεί ως βάση για την 1η εργασία
των Μεταγλωττιστών (αναγνώριση κειμένου μέσω αυτομάτου DFA).

ΠΡΟΣΟΧΗ: Προσθήκες στον κώδικα επιτρέπονται μόνο
στα σημεία (A), (B) και (Γ) - διαβάστε τα σχόλια!
'''

transitions = {
    's0': {'0': 's1', '1': 's1', '2': 's1',
           '3': 's1', '4': 's1', '5': 's1', '6': 's1', '7': 's1', '8': 's1', '9': 's1', '.': 's2',
           's1': {'0': 's1', '1': 's1', '2': 's1',
                  '3': 's1', '4': 's1', '5': 's1', '6': 's1', '7': 's1', '8': 's1', '9': 's1', '.': 's2',
                  's2': {'0': 's3', '1': 's3', '2': 's3',
                         '3': 's3', '4': 's3', '5': 's3', '6': 's3', '7': 's3', '8': 's3', '9': 's3',
                         's3': {'0': 's3', '1': 's3', '2': 's3',
                                '3': 's3', '4': 's3', '5': 's3', '6': 's3', '7': 's3', '8': 's3', '9': 's3',
                                's4': {'0': 's5', '1': 's5', '2': 's5',
                                       '3': 's5', '4': 's5', '5': 's5', '6': 's5', '7': 's5', '8': 's5', '9': 's5',
                                       's5': {'0': 's3', '1': 's5', '2': 's5',
                                              '3': 's5', '4': 's5', '5': 's5', '6': 's5', '7': 's5', '8': 's5', '9': 's5'}}}}}}

accepts = {
    's2': 'FLOAT_TOKEN',
    's3': 'FLOAT_TOKEN',
    's5': 'FLOAT_TOKEN'
}

def get_char(text, pos):
    """ Returns char (or char category) at position 'pos' of 'text',
        or None if out of bounds. """
    if pos < 0 or pos >= len(text): return None
```

The terminal output shows the results of testing the DFA with various inputs:

```
root@kali: ~
File Edit View Search Terminal Help

Testing '1234'
Result: None

Testing '.'
Result: None

root@kali: ~# python3 dfa.py
Testing '12.456'
Result: {'token': 'FLOAT_TOKEN', 'lexeme': '12.456'}

Testing '6789.'
Result: {'token': 'FLOAT_TOKEN', 'lexeme': '6789.'}

Testing '.66998'
Result: {'token': 'FLOAT_TOKEN', 'lexeme': '.66998'}

Testing '1234'
Result: None

Testing '.'
Result: None

root@kali: ~#
```